

BÀI TẬP LẬP TRÌNH MẠNG

Cơ bản

Một số command-line hữu ích

Command-line	Diễn giải
Ipconfig	The default is to display only the IP address, subnet mask and default gateway for each adapter bound to TCP/IP. >ipconfig /? Để xem chi tiết
Netstat	Displays protocol statistics and current TCP/IP network connections. >netstat /?
Ping	Ping the specified host >ping /?
Nslookup	look up a host >nslookup /?

NetworkInterface class

Mục đích: This class represents a Network Interface made up of a name, and a list of IP addresses assigned to this interface. It is used to identify the local interface on which a multicast group is joined.

Code

```
import java.net.NetworkInterface;
import java.net.SocketException;
import java.util.Collections;
import java.util.Enumeration;

public class NetworkInterfaceDemo {
    public static void main(String[] args) {
        try {
            Enumeration<NetworkInterface> interfaceEnum =
                NetworkInterface.getNetworkInterfaces();
            System.out.printf("Name \t Display name\n");
            for(NetworkInterface element :
                Collections.list(interfaceEnum)) {
                System.out.printf("%-8s %-32s\n",
                    element.getName(), element.getDisplayName());
            }
        } catch (SocketException ex) {
            ex.printStackTrace();
        }
    }
}
```

Đối tượng InetAddress

Là lớp biểu diễn cho một địa chỉ Internet Protocol (IP)

```
package vovanhai.wordpress.com;
import java.net.InetAddress;
public class TestInetAddress {
    public static void main(String[] args) throws Exception{
        InetAddress names[] =
            InetAddress.getAllByName("www.vnexpress.net");
        for(InetAddress element : names) {
            System.out.println(element);
        }
        System.out.println("-----");
        displayInetAddressInformation(names[0]);
    }

    private static void displayInetAddressInformation(
        InetAddress address) {
        System.out.println(address);
        System.out.println("CanonicalHostName: " +
            address.getCanonicalHostName());
        System.out.println("HostName: " + address.getHostName());
        System.out.println("HostAddress: " +
            address.getHostAddress());
    }
}
```

```
<terminated> TestInetAddress [Java Application]
www.vnexpress.net/111.65.248.132
-----
www.vnexpress.net/111.65.248.132
CanonicalHostName: 111.65.248.132
HostName: www.vnexpress.net
HostAddress: 111.65.248.132
```

URL/URI/URN

Sử dụng đối tượng URI và các phương thức của nó

```
package vovanhai.wordpress.com;

import java.net.URI;

public class TestURI {
    public static void main(String[] args) throws Exception{
        URI uri = new
            URI("https://www.packtpub.com/books/content/support");
        displayURI(uri);
    }

    private static void displayURI(URI uri) {
        System.out.println("getAuthority: " + uri.getAuthority());
        System.out.println("getScheme: " + uri.getScheme());
        System.out.println("getSchemeSpecificPart: "
            + uri.getSchemeSpecificPart());
        System.out.println("getHost: " + uri.getHost());
        System.out.println("getPath: " + uri.getPath());
        System.out.println("getQuery: " + uri.getQuery());
        System.out.println("getFragment: " + uri.getFragment());
        System.out.println("getUserInfo: " + uri.getUserInfo());
        System.out.println("normalize: " + uri.normalize());
    }
}
```

```
}
```

Sử dụng đối tượng URL

```
package vovanhai.wordpress.com;
import java.net.URL;
public class TestURL {
    public static void main(String[] args) throws Exception{
        URL url = new URL("https://www.packtpub.com:80/books/content/support");
        displayURL(url);
    }

    private static void displayURL(URL url) {
        System.out.println("URL: " + url);
        System.out.printf(" Protocol: %-32s Host: %-32s\n",
            url.getProtocol(),url.getHost());
        System.out.printf(" Port: %-32d Path: %-32s\n",
            url.getPort(),url.getPath());
        System.out.printf(" Reference: %-32s File: %-32s\n",
            url.getRef(),url.getFile());
        System.out.printf(" Authority: %-32s Query: %-32s\n",
            url.getAuthority(),url.getQuery());
        System.out.println(" User Info: " + url.getUserInfo());
    }
}
```

Retrieve data từ một URL

Phương thức sau đây dùng để download một tài nguyên được chỉ định lưu xuống một file

```
public long download(URL url, String destinationFilePath) throws Exception{
    long bytes=0;
    FileOutputStream fos= new FileOutputStream(destinationFilePath);
    int len=512;
    InputStream is=url.openStream();
    byte[]buffer=new byte[512];
    while(is.available()!=0) {
        len=is.read(buffer);
        bytes+=len;
        fos.write(buffer,0,len);
    }
    fos.close();
    return bytes;
}

public static void main(String[] args) {
    try {
        DownloadResource dlr=new DownloadResource();
        URL url=new
        URL("http://cdn2.tstatic.net/tribunnews/foto/bank/images/Shaila-Sabt.jpg");
        String destinationFilePath="c:/a/girl.jpg";
        long bytes=dlr.download(url, destinationFilePath);
        System.out.printf("%d bytes downloaded",bytes);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Đối tượng URLConnection

Giới thiệu

Lớp abstract URLConnection là lớp superclass của tất cả các lớp biểu diễn cho kết nối tương tác (2 chiều) giữa ứng dụng và một URL.

Để tạo được một URLConnection từ một URL ta phải có các bước sau

<code>openConnection()</code>	<code>connect()</code>
Manipulate parameters that affect the connection to the remote resource.	Interact with the resource; query header fields and contents.

----->
time

1. The connection object is created by invoking the `openConnection` method on a URL.
2. The setup parameters and general request properties are manipulated.
3. The actual connection to the remote object is made, using the `connect` method.
4. The remote object becomes available. The header fields and the contents of the remote object can be accessed

The setup parameters are modified using the following methods:

- `setAllowUserInteraction`
- `setDoInput`
- `setDoOutput`
- `setIfModifiedSince`
- `setUseCaches`

Đọc thông tin từ một host cũng như nội dung của nó

```
package vovanhai.wordpress.com;
import java.io.InputStream;
import java.net.URL;
import java.net.URLConnection;

public class SampleURLConnection {
    public static void main(String[] args) throws Exception{
        URL url=new URL("http://localhost");
        URLConnection con=url.openConnection();
        con.setDoInput(true);
        con.setDoOutput(true);
        con.connect();

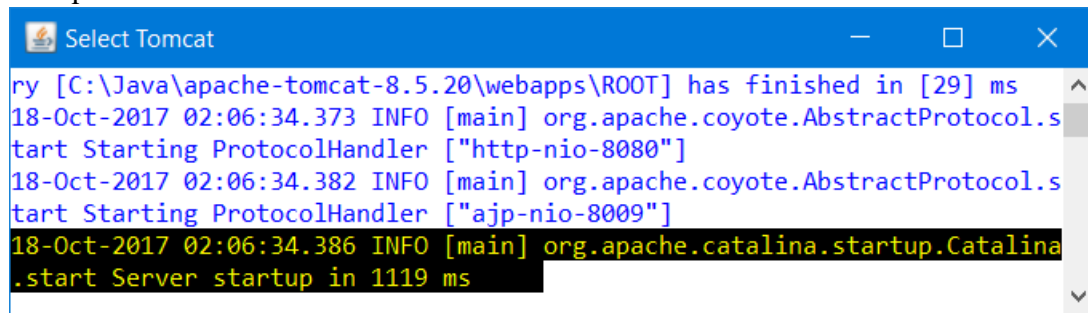
        System.out.println("length:"+con.getContentLength());
        InputStream is = con.getInputStream();
        byte []buf=new byte[512];
        while(is.available()!=-0) {
            int l=is.read(buf);
            System.out.println(new String(buf,0,l));
        }
        is.close();
    }
}
```

Gửi request đến một trang web xử lý sau đó nhận kết quả

1. Vào trang <http://tomcat.apache.org/> download Tomcat Web Server về, giải nén vào thư mục nào đó (giả sử là C:\Java\apache-tomcat-8.5.20)

2. Thêm biến môi trường JAVA_HOME
3. Vào thư mục bin của tomcat, chạy file startup.bat.

Kết quả



```
ry [C:\Java\apache-tomcat-8.5.20\webapps\ROOT] has finished in [29] ms
18-Oct-2017 02:06:34.373 INFO [main] org.apache.coyote.AbstractProtocol.s
tart Starting ProtocolHandler ["http-nio-8080"]
18-Oct-2017 02:06:34.382 INFO [main] org.apache.coyote.AbstractProtocol.s
tart Starting ProtocolHandler ["ajp-nio-8009"]
18-Oct-2017 02:06:34.386 INFO [main] org.apache.catalina.startup.Catalina
.start Server startup in 1119 ms
```

4. Vào thư mục TOMCAT_HOME\ webapps, tạo thư mục **myweb**.
5. Tạo file **sample.jsp** trong thư mục **myweb** có nội dung sau

```
<%
    String us=request.getParameter("user");
    String psw=request.getParameter("password");
    if(us.equals(psw)) {
        out.println("Welcome "+us);
    }
    else{
        out.println("Something wrong with your username & password");
    }
%>
```

6. Mở trình duyệt thử với address sau
 - a. <http://localhost:8080/myweb/sample.jsp?user=teo&password=teo>
 - b. <http://localhost:8080/myweb/sample.jsp?user=teo&password=other>
7. Viết code dùng URLConnection gửi dữ liệu đến web server

```
package vovanhai.wordpress.com;

import java.io.OutputStream;
import java.net.URL;
import java.net.URLConnection;
import java.util.Scanner;

public class SendOutputSample {
    public static void main(String[] args) throws Exception{
        String spec="http://localhost:8080/myweb/sample.jsp";
        URL url=new URL(spec);
        URLConnection urlcon=url.openConnection();
        urlcon.setDoInput(true);
        urlcon.setDoOutput(true);
        urlcon.connect();

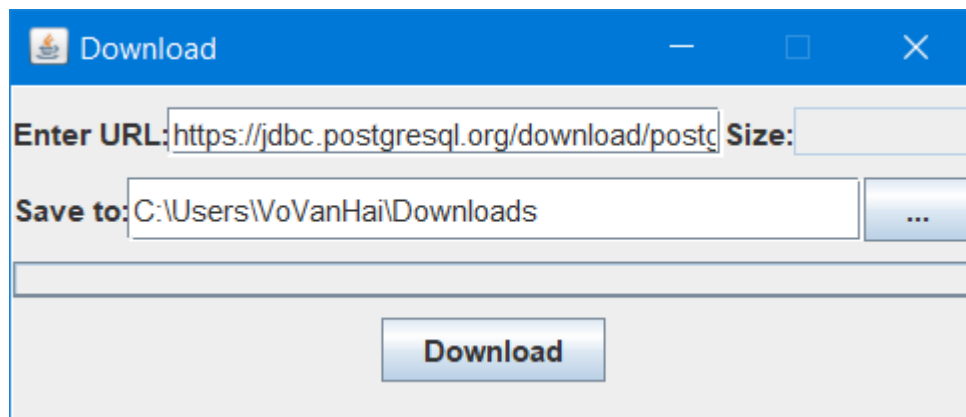
        try(OutputStream os = urlcon.getOutputStream()){
            os.write("user=teo&password=teo".getBytes());
            os.flush();
        }

        try(Scanner in=new Scanner(urlcon.getInputStream())){
            if(in.hasNextLine()) {
                String line=in.nextLine();
                System.out.println(line);
            }
        }
    }
}
```

8. Thực thi, quan sát kết quả
9. Thay password trong chuỗi tô màu vàng code trên với password bằng một giá trị gì đó khác “teo”. Quan sát kết quả.

Đối tượng HttpURLConnection/ HttpsURLConnection

Chương trình sau download một tài nguyên xác định trên mạng. Giao diện như hình



Trong quá trình download, thanh tiến trình sẽ hiển thị % downloaded

```
package vovanhai.wordpress.com;

import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.swing.Box;
import javax.swing.JButton;
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JProgressBar;
import javax.swing.JTextField;
import javax.swing.SwingWorker;

public class DownloadUsingURLConnection extends JFrame implements ActionListener{
    private JProgressBar pBar;
    private JButton btnDownload;
    private JTextField tfURL;
    private JTextField tfSize;
    private JTextField tfFolder;
    private JButton btnSelectFolder;

    public DownloadUsingURLConnection() {
        setTitle("Download");
        setSize(400, 170);setResizable(false);
        setDefaultCloseOperation(DISPOSE_ON_CLOSE);
        Box b=Box.createVerticalBox();

        Box b1=Box.createHorizontalBox();
        b1.add(new JLabel("Enter URL:"));
```



```

        b1.add(tfURL=new JTextField(20));
tfURL.setText("https://jdbc.postgresql.org/download/postgresql-9.2-1002.jdbc4.jar");
        b1.add(new JLabel("Size:")); b1.add(tfSize=new
JTextField(5));tfSize.setEditable(false);

        Box b2=Box.createHorizontalBox();
        b2.add(new JLabel("Save to:")); b2.add(tfFolder=new JTextField(20));
        String homeDownload =
System.getProperty("user.home")+File.separator+"Downloads";
        tfFolder.setText(homeDownload);
        b2.add(btnSelectFolder=new JButton("..."));
        btnSelectFolder.addActionListener(this);

        Box bButton=Box.createHorizontalBox();
        bButton.add(btnDownload=new JButton("Download"));
        btnDownload.addActionListener(this);

        b.add(Box.createVerticalStrut(8));b.add(b1);
        b.add(Box.createVerticalStrut(8)); b.add(b2);

        b.add(Box.createVerticalStrut(8));
        b.add(pBar=new JProgressBar(),BorderLayout.NORTH);
        b.add(Box.createVerticalStrut(8));
        b.add(bButton);

        this.add(b, BorderLayout.NORTH);
    }

    private void trustAllHosts(){
        // Create a trust manager that does not validate certificate chains
        TrustManager[] trustAllCerts = new TrustManager[] { new
X509TrustManager(){
            public java.security.cert.X509Certificate[] getAcceptedIssuers(){
                return new java.security.cert.X509Certificate[] {};
            }
            public void checkClientTrusted(X509Certificate[] chain, String
authType) throws CertificateException{
            }
            public void checkServerTrusted(X509Certificate[] chain, String
authType) throws CertificateException{
            }
        } };
        // Install the all-trusting trust manager
        try{
            SSLContext sc = SSLContext.getInstance("TLS");
            sc.init(null, trustAllCerts, new java.security.SecureRandom());
            HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
        } catch (Exception e){
            e.printStackTrace();
        }
    }

    private HttpURLConnection getConnection(URL url) throws Exception{
        HttpURLConnection connection=null;
        if (url.getProtocol().toLowerCase().equals("https")){
            trustAllHosts();
            HttpsURLConnection https = (HttpsURLConnection)
url.openConnection();
            connection = https;
        } else
            connection = (HttpURLConnection) url.openConnection();
        //con.setRequestProperty("Accept-Encoding", "identity");
    }

```

```

        connection.connect();
        return connection;
    }

    @Override
    public void actionPerformed(ActionEvent ae) {
        Object o=ae.getSource();
        if(o.equals(btnDownload)) {
            try {
                //URL url=new
                URL("https://wallpapercave.com/wp/Xi3z1EN.jpg");
                //URL url=new URL("http://localhost/hello/a.zip");

                pBar.setValue(0);
                String urlspec=tfURL.getText();
                String destFolder=tfFolder.getText();//check???

                URL url=new URL(urlspec);
                HttpURLConnection con=getURLConnection(url);
                String size=con.getContentLength()+"";
                tfSize.setText(size);

                DownloadURL qlu=new DownloadURL(con,pBar,destFolder);
                qlu.execute();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        if(o.equals(btnSelectFolder)) {
            JFileChooser j = new JFileChooser();
            j.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
            if( j.showSaveDialog(null)==JFileChooser.APPROVE_OPTION) {
                tfFolder.setText(j.getSelectedFile().getAbsolutePath());
            }
        }
    }

    public static void main(String[] args) {
        new DownloadUsingURLConnection().setVisible(true);
    }
}

class DownloadURL extends SwingWorker<Void, Long>{
    private final int BUFFER_SIZE = 4096;
    private JProgressBar progress;
    private HttpURLConnection urlCon;
    private String destFolder;

    public DownloadURL(HttpURLConnection urlCon, JProgressBar progress,String
destFolder) {
        this.progress = progress;
        this.urlCon=urlCon;
        this.destFolder=destFolder;
    }

    @Override
    protected Void doInBackground() throws Exception {
        downloadFile(urlCon,destFolder);
        return null;
    }

    public void downloadFile(HttpURLConnection conn, String saveDir)throws Exception
{
        String fileURL=conn.getURL().toExternalForm();

```

```

        int responseCode = conn.getResponseCode();

        // always check HTTP response code first
        if (responseCode == HttpURLConnection.HTTP_OK) {
            String fileName = "";
            String disposition = conn.getHeaderField("Content-Disposition");

            if (disposition != null) {
                // extracts file name from header field
                int index = disposition.indexOf("filename=");
                if (index > 0) {
                    fileName = disposition.substring(index + 10,
                        disposition.length() - 1);
                }
            } else {
                // extracts file name from URL
                fileName = fileURL.substring(fileURL.lastIndexOf("/") + 1,
                    fileURL.length());
            }
            int length=urlCon.getContentLength();
            int downloaded = 0;
            // opens input stream from the HTTP connection
            String saveFilePath = saveDir + File.separator + fileName;

            InputStream inputStream = urlCon.getInputStream();

            // opens an output stream to save into file
            FileOutputStream outputStream = new FileOutputStream(saveFilePath);

            int bytesRead = -1;
            int pos=0;
            byte[] buffer = new byte[BUFFER_SIZE];

            while ((bytesRead = inputStream.read(buffer))!=-1) {
                outputStream.write(buffer, 0, bytesRead);
                downloaded+=bytesRead;
                pos=(downloaded * 100) / length;
                progress.setValue(pos);
                //System.out.println(pos+"\t"+downloaded);
            }
            outputStream.close();
            inputStream.close();
        } else {
            JOptionPane.showMessageDialog(null,"No file to download. Server
replied HTTP code: " + responseCode,
                "Download Error",JOptionPane.ERROR_MESSAGE);
        }

        @Override
        protected void done() {
            System.out.println("File downloaded");
        }
    }
}

```

TCP Socket

Ví dụ 1

Sử dụng Socket class

Ví dụ sau sử dụng Java Socket truy xuất đến một trang web

```
import java.io.InputStream;
import java.io.OutputStream;
import java.io.PrintWriter;
import java.net.Socket;

public class ConnectHttpUsingSocket {
    public static void main(String[] args) throws Exception{
        //connect
        Socket socket=new Socket("localhost", 80);
        //send request
        OutputStream os = socket.getOutputStream();
        PrintWriter out=new PrintWriter(os,true);
        out.println("GET / HTTP/1.0");
        out.println();
        out.flush();
        //receive data
        InputStream is = socket.getInputStream();
        int len=0;
        byte []buffer=new byte[4086];
        while((len=is.read(buffer))!=-1) {
            String data=new String(buffer,0,len);
            System.out.println(data);
        }
        //close connection
        socket.close();
    }
}
```

Sử dụng ServerSocket class

Ví dụ sau tạo 1 server lắng nghe trên cổng 999 chờ bất kỳ 1 client nối đến. Khi client nối và gửi 1 chuỗi request thì sẽ đảo ngược chuỗi đó sau đó gửi kết quả lại cho client.

Server

```
package vovanhai.wordpress.com.soc;

import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Scanner;

public class ReverseStringServer {
    public static void main(String[] args) throws Exception{
        System.out.println("Server was listened on port 9999");
        ServerSocket svr=null;
        try {
            //boud server on port
            svr=new ServerSocket(9999);
            boolean state=true;
            while(state) {
                //accept any client
                //client's socket-proxy was saved in socket variable
                Socket socket = svr.accept();
                //read client request -sample: text request
                Scanner sc=new Scanner(socket.getInputStream());
                String request=sc.nextLine();
            }
        }
    }
}
```

```

        //processing request
        String response=new
StringBuilder(request).reverse().toString();
        //send response to client - sample: text response
        PrintWriter out=new PrintWriter(
            socket.getOutputStream(),true/*auto-flush*/);
        out.println(response);

        //close socket
        out.close();
        sc.close();
        socket.close();
    }

    } catch (Exception e) {
        e.printStackTrace();
    }
    finally {
        if(svr!=null)
            svr.close();
    }

}
}

```

Client

```

package vovanhai.wordpress.com.soc;

import java.io.PrintWriter;
import java.net.Socket;
import java.util.Scanner;

public class ReverseStringClient {
    public static void main(String[] args) throws Exception{
        Socket soc=new Socket("localhost", 9999);
        //send request
        PrintWriter out=new PrintWriter(soc.getOutputStream(),true);
        out.println("This is sample request test");
        //read response
        Scanner in=new Scanner(soc.getInputStream());
        String res=in.nextLine();
        //process
        System.out.println(res);
        //close resources
        in.close();
        out.close();
        soc.close();
    }
}

```

Ví dụ 2

Ví dụ 3

UDP Socket

Java NIO2

The Sockets APIs

TCP Server/Client Applications

Blocking TCP Server/Client Application

Ví dụ sau đây sử dụng channel để tạo 1 server lắng nghe trên cổng 5000 chờ bất kỳ kết nối nào gửi yêu cầu đến, sau đó lấy thời gian hệ thống trả về

Server

```
package vovanhai.wordpress.com.channel;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.nio.ByteBuffer;
import java.nio.channels.ServerSocketChannel;
import java.nio.channels.SocketChannel;
import java.util.Date;

public class ServerSocketChannelTimeServer {
    public static void main(String[] args) {
        System.out.println("Time Server started");
        try {
            //bind socket on port 5000
            ServerSocketChannel serverSocketChannel =
ServerSocketChannel.open();
            serverSocketChannel.socket().bind(new InetSocketAddress(5000));

            while (true) {
                System.out.println("Waiting for request ...");
                //accept any connecting request
                SocketChannel socketChannel = serverSocketChannel.accept();
                if (socketChannel != null) {
                    String dateAndTimeMessage = "Date: " + new
Date(System.currentTimeMillis());
                    ByteBuffer buf = ByteBuffer.allocate(64);
                    buf.put(dateAndTimeMessage.getBytes());
                    buf.flip();
                    //send response
                    while (buf.hasRemaining()) {
                        socketChannel.write(buf);
                    }
                    System.out.println("Sent: " + dateAndTimeMessage);
                }
            }
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
}
```

Client

```
package vovanhai.wordpress.com.channel;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.SocketAddress;
import java.nio.ByteBuffer;
import java.nio.channels.SocketChannel;

public class SocketChannelTimeClient {
```

```

public static void main(String[] args) {
    //connect to server
    SocketAddress address = new InetSocketAddress( "127.0.0.1", 5000);
    try (SocketChannel socketChannel = SocketChannel.open(address)) {
        //read response
        ByteBuffer byteBuffer = ByteBuffer.allocate(64);
        int bytesRead = socketChannel.read(byteBuffer);
        while (bytesRead > 0) {
            byteBuffer.flip();
            while (byteBuffer.hasRemaining()) {
                System.out.print((char) byteBuffer.get());
            }
            System.out.println();
            bytesRead = socketChannel.read(byteBuffer);
        }
    } catch (IOException ex) {
        ex.printStackTrace();
    }
}

```

Non-Blocking TCP Client/Server Application

UDP Server/Client Applications

The Asynchronous Channel API

Bài tập

Bài 1

Dựa vào phần hướng dẫn “Đối tượng HttpURLConnection/ HttpsURLConnection” viết một chương trình download manager với yêu cầu:

- Giao diện quản lý lịch sử download
- Có thể download nhiều tài nguyên cùng lúc
- Bổ sung thêm có thể download với giao thức ftp

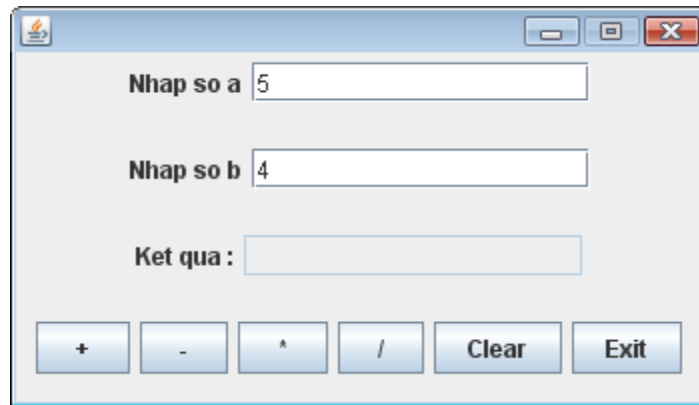
Bài 2

Viết 1 chương trình có tên Calculator_server nhận 1 biểu thức gồm 2 chữ số và 1 phép toán sau đó thực thi biểu thức này và gửi kết quả lại cho client.

Sửa chữa chương trình cho phép nhiều client kết nối cùng lúc.

Phía client, viết giao diện gồm 2 JTextField cho việc nhập số, 1 JLabel cho việc xuất kết quả. Các nút Cộng, trừ, nhân, chia, clear và thoát.

Giao diện cho client như sau:



Bài 3

Viết 1 server cho phép nhiều client kết nối cùng lúc với các yêu cầu sau:

Client có thể gửi yêu cầu là đường dẫn đến 1 ổ đĩa hoặc 1 thư mục nào đó bất kỳ trên server. Nếu đường dẫn đó tồn tại thì sẽ gửi về danh sách các thư mục con và các tập tin trong ổ đĩa / đường dẫn đó.

Thiết kế client với cơ chế GUI nhận kết quả từ server và biểu diễn kết quả nhận được lên 1 JTree.

Hướng dẫn:

Đoạn code sau liệt kê tất cả thư mục, tập tin trong 1 đường dẫn path cho trước rồi đưa vào 1 đối tượng ArrayList

```
File f=new File(path);
ArrayList<File>lstFiles=null;
if(f.exists() && f.isDirectory()) {
    lstFiles=new ArrayList<File>();
    File []files=f.listFiles();
    for(File x:files)
        lstFiles.add(x);
}
```

Đoạn code sau tạo luồng có thể chuyển 1 đối tượng tới client

```

OutputStream os = income.getOutputStream();
ObjectOutputStream oos=new ObjectOutputStream(os);
oos.writeObject(lstFiles);
oos.flush();

```

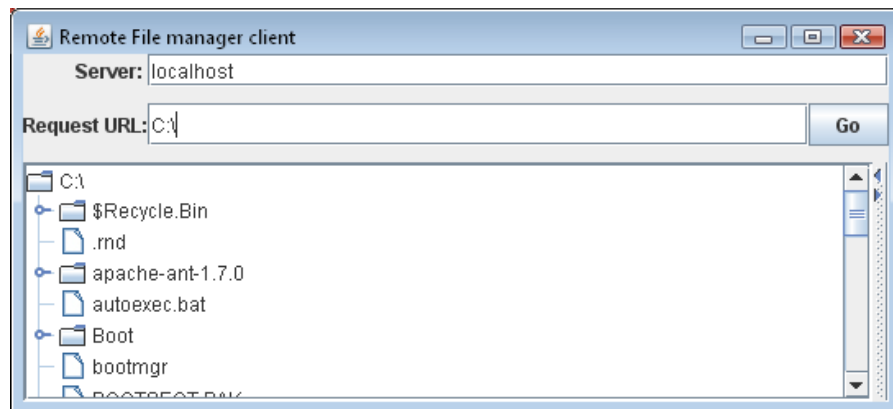
Đoạn code sau đây nhận đối tượng từ server

```

ObjectInputStream ois=new ObjectInputStream(soc.getInputStream());
ArrayList<File>list=(ArrayList<File>)ois.readObject();

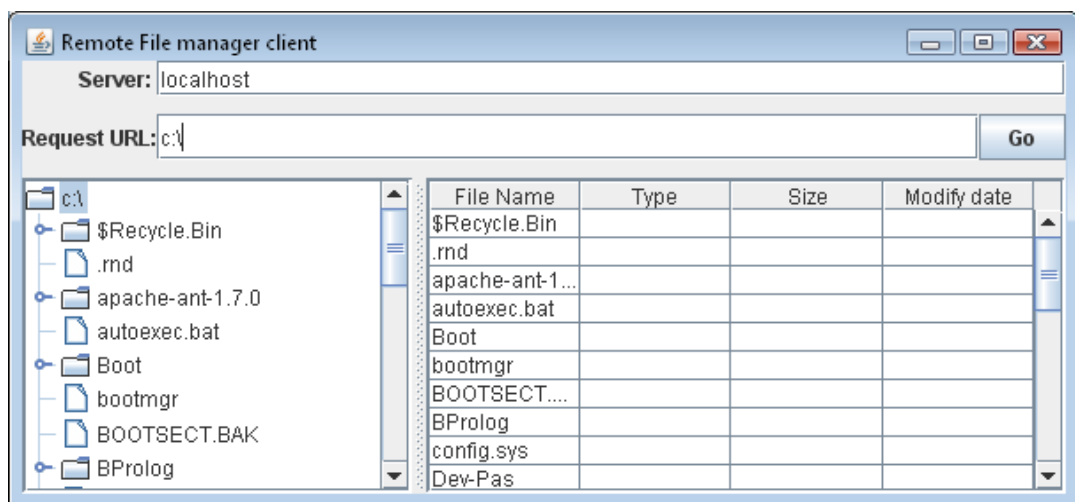
```

Giao diện phía client như sau:



Bài 4

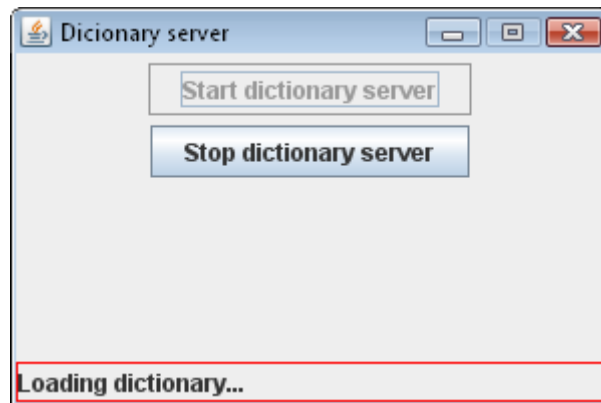
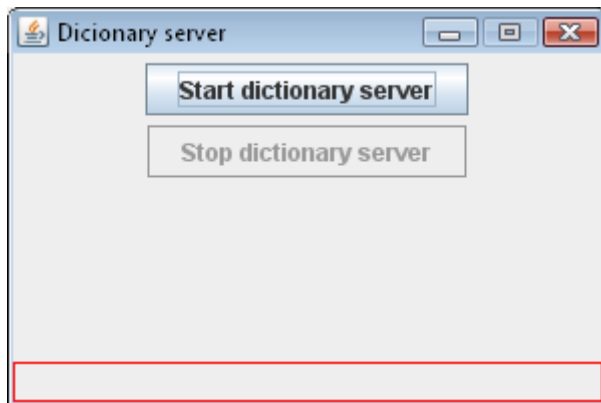
Cải tiến bài 2 thành 1 chương trình quản lý thư mục từ xa và cho phép các thao tác cơ bản như download, delete files / folders.



Bài 5

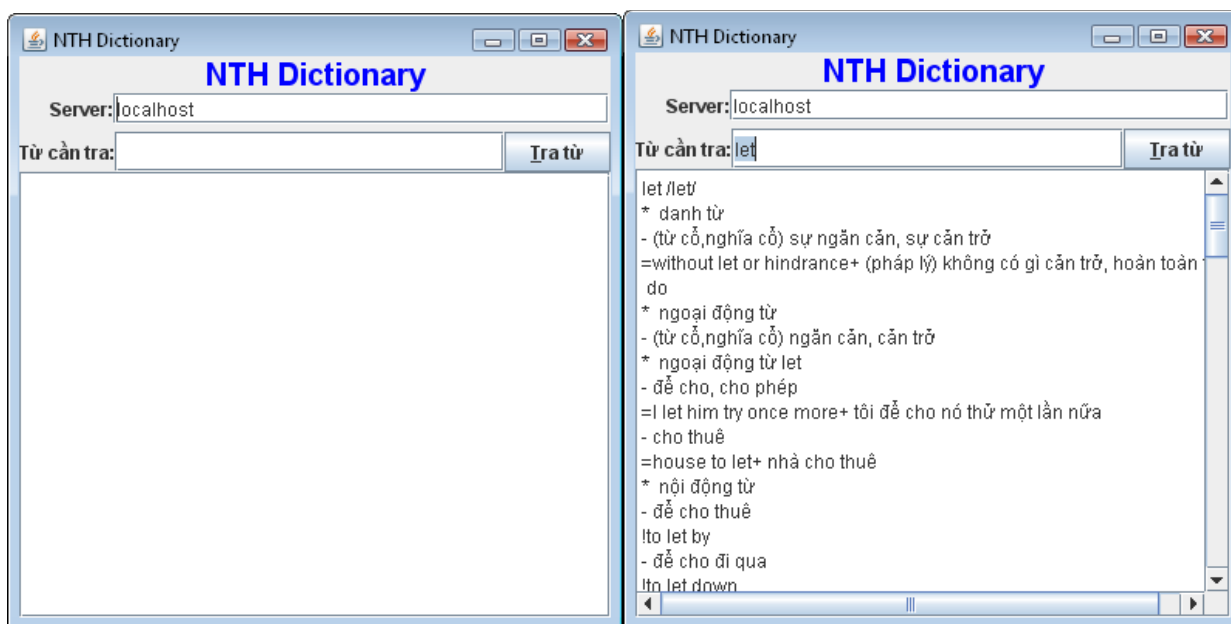
Viết 1 chương trình từ điển cho phép tra từ qua mạng. Việc tra từ này phải đảm bảo nhiều người có thể tra cùng lúc. Việc thiết kế chương trình gồm 2 phần: Phần server và phần client.

Phần server: chỉ sẽ được chạy trên server có giao diện như sau:



Khi người dùng start server từ điển, server này sẽ lắng nghe trên cổng 2520 và sẽ nhận đầu vào là từ cần tra sau đó thực hiện việc tra từ và trả kết quả về cho client hoặc là 1 đối tượng từ đã tra trong trường hợp 745 tra thành công, hoặc là null nếu từ không tồn tại.

Phần Client: sẽ được triển khai ở phía client, có giao diện như sau



Bài 6

Viết chương trình giả lập 1 chat room. Người dùng nhập địa chỉ server, tên nick chat vào và có thể chat cùng nhau trên chat room này.

